edited by
JAMES P. BIRK
Arizona State University, Tempe, AZ 85281

An Inexpensive Computer Station for Undergraduate Laboratories Using the Atari 800XL

Edwin F. Meyer
DePaul University, Chicago, IL 60614

It seems generally agreed among chemical educators that computers in undergraduate teaching labs are a good thing as long as their primary purpose is the elimination of tedium and they do not "mystify" as they simplify. Virtually every student who completes a degree in chemistry and gains employment as a chemist will run into a situation where a computer is being used to collect data and/or control equipment, and there should be some exposure in the educational process to the basic principles involved.

The kind of "interfacing" that has been emphasized in chemical education thus far in this country has been largely limited to using the "game paddle inputs" of a home computer, which allows the connection of any device that looks like a variable resistor to the computer. This approach has served admirably as an introduction to the power and versatility of inexpensive home computers as data collectors and handlers but suffers from significant disadvantages. The most obvious is the limitation to 8 bits of information; one would like to be able to obtain better precision than this provides (at half scale we can expect roughly 1% reproducibility). Another is the requirement that the resistance of the transducer used be consistent with that of the game paddle it replaces.

It is possible, without spending inordinate sums of money, to convert one of these home computers into a research-grade instrument with a resolution of 1 bit in 4096, if one knows a little about digital electronics. This article describes an interface for the Atari 800XL computer based on a 12-bit analogue-to-digital converter (ADC). We have incorporated six of them into "computer stations" in our upper track freshman laboratory. In general, the variables in question (e.g., temperature vs. time for coffee cup calorimeter experiments, pH vs. volume titrant) are plotted in real time on the monitor screen, and after collection of the data, a hard copy of the plot is produced on a printer, along with a table of the data. We use similar stations in our physical chemistry laboratory, where more sophisticated curve-fitting routines are included.

Computers in the Undergraduate Laboratory

Our philosophy has been that the chemistry major should perform one or more computer-interfaced experiments, and be given the *opportunity* to learn as much about the interfacing aspects thereof as desired without the *requirement* actually to "do" any interfacing.

A problem with putting computers into undergraduate laboratories is the expense involved. There should not be more than two students per keyboard if each is to benefit from the experience. The IBM clones, for which hardware and software for interfacing are readily available, start at about $1000; interfacing capability adds several hundred dollars more, and, after a monitor and printer are included, the total cost is pushing $2000 per unit.

The Atari 800XL computer, with resident BASIC and 64K of memory, while no longer being manufactured, is currently available from "resale" suppliers for less than $70. A disk drive (Model 1050) plus DOS costs about $120, and a TV monitor can be picked up for less than $50. For plotting data we have used the 1020 four-color printer, which costs $30, but more recently find that transferring data to IBM-compatible disks for use with "spreadsheet" software is an attractive alternative. (See General Comments at end of article.) Including parts for the interface our total cost for hardware has been about $400 per unit.

General Description

The "interface" for the Atari, described herein, is readily put together using a 12-bit ADC and three latches and is connected to the computer through the two game ports. The first game port is used to carry 4 bits of data into the computer; the second is used to monitor the ADC (i.e., to ensure that an analogue-to-digital conversion is complete before trying to "read" the data on the 12 output pins of the ADC) and to control the flow of 12 bits of data, four at a time, from each of the three latches, into the Atari.

When using the Atari to collect data (analogue voltage), a main program written in BASIC calls an assembly language subroutine that "reads" the output of the ADC chip. All the "cosmetics" of the particular experiment are contained in the BASIC program. For example, in a cooling curve experiment, the voltage drop across a thermistor bridge is digitized by the ADC chip, read directly as "bits" by the subroutine, and stored in memory. The BASIC program converts the raw data to temperature using a calibration equation, records the time at which the temperature was read, plots the point on the TV monitor, and saves the data for a subsequent dump to the printer, where the time–temperature data are listed, and a plot of the cooling curve is produced.

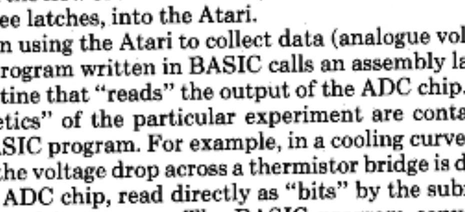

Figure 1. One of the two game ports on the Atari 800XL. Pins 1 through 4 of each port provide access to ½ byte of memory; pin 1 of port 1 is the LSB, pin 4 of port 2, the MSB of the byte that resides at memory location 54016(decimal). Pin 7 is +5 V; pin 8 is ground.

The two game ports on the Atari are made up of nine pins each (see Fig. 1). In each port, pin 8 is used for electrical ground (to which the ground pin of each of the chips of the interface should be connected), and four are data lines directly connected to a specific memory location (54016 decimal). Those in port 1 access the four lower bits, and those in port 2 access the four higher bits of the byte that resides at that location. The other pins are not used in our application and need not concern us here.

Fortunately, these eight data line pins (total) can be set individually for either input or output by simply storing a certain number in a nearby memory location (this is one of the functions of a PIA, or "peripheral interface adapter", chip that Atari uses in its game port circuitry). In our use of the ports, we want three pins to be used for output to tell each of the latches (see below) when to send its data and five to be used for input—one to monitor the BUSY pin on the ADC chip (see below) and four to carry data.

A potential problem with reading 12 bits of data into the computer four bits at a time is the possibility that the datum in question *changes* during the time interval required for reading. In our applications this is never a problem since our ADC chip takes about 160 ms per conversion and the reading process is completed in less than 1 ms. (Faster converters are available, but at higher cost; ours costs less than $20.)

Because the computer is generally ready to read the ADC chip well before it has completed a conversion, the latter is equipped with a BUSY pin, which maintains a high status until the conversion is complete, at which time the pin "goes low". The fifth input line is connected to this pin, and a loop in the assembly language subroutine monitors its status to postpone attempts to read data before they are present and stable on the output pins of the ADC chip.

So reading one analogue voltage involves (1) waiting for the BUSY pin to go low, (2) telling the first latch to send the bits on its four lines into port 1, (3) sending this information to a certain memory location, and (4) repeating the second and third steps for the other two latches, using different memory locations for each. After the subroutine returns control to BASIC, the decimal representation of the number of bits corresponding to the voltage read is obtained (using the PEEK command on the memory locations used for the latch contents) by taking the sum of 256 times the highest latch contents, 16 times that of the next, and one times that of the lowest latch. At this point the information represented by the original analogue voltage signal is in the computer's memory and can be manipulated in whatever way the programmer wishes.

Hardware

The heart of the interface is an Analog Devices AD7552KN 12 bit analogue-to-digital converter, which comes with excellent documentation. Its use requires a power supply capable of ±5 V and +12 V (one made by Coleco can be bought at Radio Shack for $5); a reference voltage, for which the AD584 chip works well (when the 12-V output from the power supply is connected to one of its pins, a stable 5-V appears on another); a 1-kΩ pot; and various resistors and capacitors. Two diodes are used in a "protection" circuit (described in the documentation that comes with the AD7552 chip), whose inclusion we have found very worthwhile, and a push-button start-up switch is convenient for use to initiate the A/D conversion on power-up. In our experience the rather elaborate grounding scheme described in the documentation has not been necessary. In operation, the analogue voltage is simply connected to the chip's input pin, and its digital representation appears on 12 others.

Figure 2 provides a circuit diagram for those parts of the interface not included in the documentation provided with the ADC chip. The lines from the 12 data pins (least significant bit at pin 21; most significant bit at pin 32 of the ADC chip) are split into three groups of four and are connected to the input pins (2, 5, 9, and 12) of three 74125 "quad latch" chips (about $0.50 each). These are 14 pin chips: four input, four output, four to control exactly when the output pins are activated, and the ubiquitous power and ground pins. The control pins on each individual latch (pins 1, 4, 10, and 13) are wired together, since we want all four to "fire" at the same time (but not at the same time as the other latches, of course). One of the control pins from each latch is connected to one of the three pins in port 2 that are to be used as output. We connected pin 3 of port 2 to the latch holding the four most significant bits, pin 2 thereof to the latch with the middle four bits, and pin 1 to the latch holding the least significant four bits of data. (Pin 4 of port 2 is used to monitor the BUSY pin (pin 36) of the ADC chip, as described above.) The software described below assumes this assignment of pins in port 2.

The four output pins from each latch (3, 6, 8, and 11) are connected as follows: the three pins representing the least significant bit (pin 3 on each latch) are wired together and into pin 1 of port 1. The three pins representing the next most significant bit (pin 6 on each latch) are wired together and into pin 2 of port 1, and so on for the other two output pins on the latches. This amounts to a four-line "data bus" into port 1, and a three-line "control bus" out of port 2. The 74125 latches are designed to send out their data when the control line goes low and to "disconnect" when the control line goes high. To read the contents of the first latch, we simply put ones on the control pins of the other two latches and a zero on the control pin of the first latch, and the others are read in an analogous fashion. All this is accomplished by the assembly language program below.

Software

Atari BASIC uses the command, USR(mem), where mem represents the decimal representation of the location in memory of the first command in the assembly language subroutine, to call that subroutine. The following BASIC program serves simply to read the output of the ADC chip and print its decimal representation on the screen at every conversion:

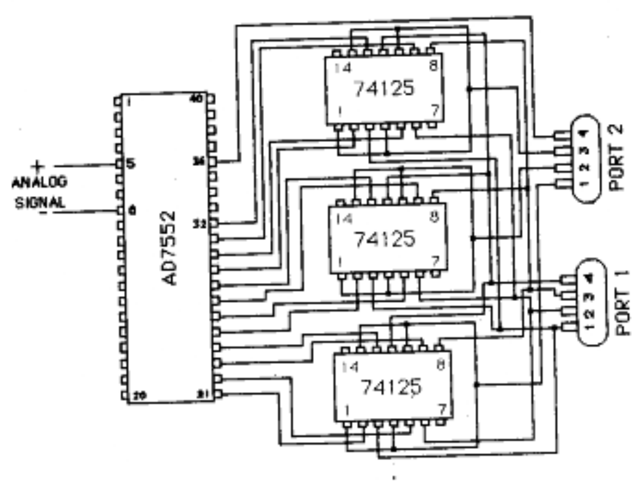

Figure 2. Circuit diagram for connecting the 12 data lines from the ADC chip to the two game ports of the Atari (see text). The analogue signal to be measured is connected across pins 5 and 8, and its digitized representation appears on pins 21 through 32 of the AD7552 ADC chip. Four bits are held in each of the three latches until the computer is ready to read them. The connections to the other pins are described in documentation provided with the chip.

```
10 REM **ASSEMBLY ROUTINE STARTS AT MEM LOC
   20480 (HEX 5000)
20 A=USR(20480)
30 BITS=256*PEEK(21760)+16*PEEK(21761)+PEEK(21762)
40 REM **ASSEMBLY ROUTINE STORES BITS IN 21760
   THRU 21762
50 REM **(HEX 5500 THRU 5502)
60 PRINT BITS
70 GOTO 20
```

The assembly language program called looks like this:

```
10        *= $5000   ;STARTS AT HEX 5000 (DEC 20480)
20        .OPT OBJ   ;STORES ASSEMBLED PROGRAM IN
                      MEM
30        PLA
40        CLD        ;INITIALIZE
50 LOOP   LDA $D300  ;LOAD ACC WITH DATA ON PORTS 1
                      AND 2
60        AND #$80   ;IGNORE ALL BUT LINE TO BUSY PIN
                      ON A/D
70        BNE LOOP   ;IF NOT ZERO, LOOP BACK TO STEP 50
80        LDA #$30   ;MAKES PORT CONTENTS ENTER DI-
                      RECTION
90        STA $D302  ;...REGISTER OF PIA CHIP
100       LDA #$70   ;MAKES BITS 4,5,6 OUTPUT,
110       STA $D300  ;...0,1,2,3 AND 7 INPUT
120       LDA #$34   ;CHANGES PORT BACK TO R/W
130       STA $D302  ;...STATUS
140       LDA #$30   ;ACTIVATES LATCH HOLDING
150       STA $D300  ;...FOUR MSB'S OF DATA
160       LDA $D300  ;READS ALL EIGHT PINS OF PORTS 1
                      AND 2
170       AND #$0F   ;IGNORES FOUR PINS OF PORT 2
180       STA $5500  ;STORES FOUR DATA BITS IN HEX 5500
190       LDA #$50   ;CLOSES FIRST LATCH, OPENS LATCH
200       STA $D300  ;...HOLDING NEXT FOUR BITS OF
                      DATA
210       LDA $D300  ;READS ALL EIGHT PINS OF PORTS 1
                      AND 2
220       AND #$0F   ;IGNORES FOUR PINS OF PORT 2
230       STA $5501  ;STORES DATA IN HEX 5501
240       LDA #$60   ;CLOSES SECOND LATCH, OPENS
                      LATCH
250       STA $D300  ;...HOLDING FOUR LSB'S OF DATA
260       LDA $D300  ;READS EIGHT PINS OF PORTS 1 AND 2
270       AND #$0F   ;IGNORES FOUR PINS OF PORT 2
280       STA $5502  ;STORES DATA IN HEX 5502
290       RTS        ;RETURNS CONTROL TO BASIC PRO-
                      GRAM
```

Once back in BASIC, the three groups of bits are collected (step 30 in the BASIC program above), converted to decimal, printed on the screen, and the process is repeated.

The interface can be used in any experiment for which the variable of interest can be converted to an analogue voltage. For example, the thermistor bridge mentioned above to measure temperature can be used to produce cooling curves, study heats of mixing, freezing point depression, etc.; a light-sensitive cell provides the basis for computerized colorimetry; pressure transducers are available so that rates of gaseous reactions may be followed or gaseous viscosity measured, etc.

General Comments

While the Atari 1020 printer provides an immediate permanent record of plots of data, the small paper width limits the precision of quantitative measurements therefrom (e.g., break points from cooling curves). If LOTUS-123 (or similar spreadsheet software) is available, the purchase of a means of transferring data from the Atari disks to IBM-compatible disks, such as the P:R:Connection ($59 from Micro-Marketing, Inc., 2400 Reach Road, Williamsport, PA 17701) is recommended. Our current procedure is to give each student a disk formatted for the computer containing LOTUS-123 (we use Zenith's) at the outset of the course. This disk is considered a part of his or her "research notebook". Immediately after each interfaced experiment, during which data are saved to Atari disks, those data are transferred to the students' personal disks (a procedure easily learned by the students) using software that comes with the P:R:Connection for the Atari, and PROCOMM for our IBM-compatible Zenith computer. Students then, at their leisure, "Import" their data into a LOTUS-123 file for subsequent graphing and data regression. (See "Spreadsheets in Physical Chemistry" by David M. Whisnant, Vol. IIB, No. 1 of *J. Chem. Educ.: Software*.)

It should be mentioned that there may be a one-time investment of about $50 for a cartridge (or disk) that allows writing and editing assembly language programs for the Atari. We have found the MAC/65 cartridge plus documentation from OSS, Inc., San Jose, CA, very satisfactory. (This purchase is not absolutely necessary since BASIC can be used to load an assembly language program into memory. Such a program for this interface is provided in the appendix. This approach is cumbersome at best, however, and it is the opinion of the author that the ability to create and edit assembly language programs conveniently is a fundamental part of interfacing and is well worth the investment of time and money.)

The 5-V reference chip (AD584) is powered by the 12-V output of the Coleco power supply. The output of this chip should *not* be used to power the latches. We connect the power pins of the latches directly to the 5-V option of the power supply.

When a 5-V reference is used, the AD7552KN is able to convert analogue voltages between zero and 2.35 V i.e., 2.35 V equals 4096 bits, and the resolution of the interface is 0.6 mV if the resolution of the chip is 1 bit. In our experience this is very nearly accurate; our reproducibility is about ±1 mV, in general.

To use the interface to best advantage, of course, its entire range should be used. If the maximum signal for an experiment is 1 V, and the maximum precision is desired, the signal should be amlified by a factor of about 2.3 before being sent to the interface. We have found the AD625 chip (an "instrumentation amplifier") from Analog Devices to be convenient here, since its gain can be user determined by choosing a pair of resistors.

The Analog Devices chips used in the interface may be obtained from one's local supplier, who can be found by writing to: Analog Devices, Two Technology Way, P.O. Box 280, Norwood, MA 02062. A supplier of many kinds of chips, e.g., the latches used in this interface, is: Jameco Electronics, 1355 Shoreway Rd., Belmont, CA 94002.

A good source for suppliers of Atari equipment and software is *Antic* magazine, available at bookstores and many discount houses.

One of the advantages of the Atari is its readily accessible internal clock. In the cooling curve experiment, for example, the temperature of the freezing mixture is required *as a function of time*. The following excerpt from a BASIC program starts a timer and takes a reading every D seconds:

```
10 REM **CHOOSE TIME INTERVAL FOR POINTS
20 PRINT"ENTER TIME INTERVAL DESIRED (IN SE-
   CONDS):"
30 INPUT D
40 REM **INITIALIZE CLOCK
50 TO=0
60 POKE 18,0:POKE 19,0:POKE 20,0
80 FOR I=1 TO N
90 REM **N IS NUMBER OF POINTS TO BE READ ON
```

```
    COOLING CURVE
100 T=INT((PEEK(18)*65536+PEEK(19)*256+PEEK(20))/
    59.92334
110 REM **CLOCK NOT QUITE 60 CYCLES!
120 IF T=TO+D THEN 150
130 REM **IF INTERVAL COMPLETE, SAVE VALUE OF
    TIME AND READ TEMP
140 GOTO 100
150 T(I)=T
160 REM **CALL ASSEMBLY LANGUAGE PROGRAM TO
    GET BITS
170 REM **ETC.
```

Conclusion

Since putting together the interfaces for our freshman upper track laboratories, we have introduced them into our physical chemistry laboratories and expect to use them in our analytical course as well. The applications of the interface are limited only by the imagination of the user and the time available for writing the software.

In the author's experience, the most difficult part of learning about computer hardware is *getting started*. The kind of book we all would like to be able to refer to for help simply does not exist. What one needs is several books, a computer to play with, and most of all, a *person* to call when nothing seems to make sense! I knew nothing about computer hardware two years ago, and the availability of such a person was the *sine qua non* of the development of the interface described herein. Those who "get stuck" working on it should feel free to call me at (312) 341-8180.

Appendices

The assembly language program (see text) which reads the output of the ADC chip may be loaded from BASIC (i.e., without purchasing an assembler cartridge) using the following program:

```
10 CLR: DIM E(64)
20 FOR I=1 TO 64
30    READ S
40    POKE 4999+I,S
50 NEXT I
60 DATA   104,216,173,0,211,41,128,208,249,169,48,141,2,211,169,
   112,141,0,211,169,52,141,2,211,169,48,141,0,211,173
70 DATA    0,211,41,15,141,0,85,169,80,141,0,211,173,0,211,41,15,
   141,1,85,169,96,141,0,211,173,0,211,41,15,141
80 DATA 2,85,96
```

The following books have proved useful in developing the ideas involved in the interfacing project described herein:

Chadwick, I. *Mapping the Atari*; COMPUTE!: Greensboro, NC, 1985.
Coan, J. S.; Kushner, R. *Basic Atari BASIC*; Hayden: Hasbrouck Heights, NJ, 1984.
Inman, D.; Inman, K. *The Atari Assembler*; Reston: Reston, VA, 1981.
Leventhal, L. A. *6502 Assembly Language Programming*; Osborne/McGraw-Hill, Berkeley, CA, 1986.
Poole, L.; McNiff, M.; Cook, S. *Your Atari Computer*; Osborne/McGraw-Hill, Berkeley, CA, 1982.
Zaks, R. *Programming the 6502*; Sybex: Berkeley, CA, 1983.

Parts List

From Analog Devices:
- Voltage Reference Chip AD584KH
- 12-Bit ADC Chip AD7552KN

From Jameco:
- 3 × Quad Latch 74125 (The 74126 may be substituted; its control signal is opposite to the 74125, so the assembly language routine would require changing three commands.)

From Radio Shack:
- 40-pin socket
- 3 × 14-pin socket
- push-button switch
- 1-K potentiometer
- 2 × 10-K resistors
- 1.8-M resistor
- Coleco Power Supply
- 2 × 9-pin female D connector
- 0.022-mF capacitor
- 470-pF capacitor
- Schottky Diode
- IN914 Diode
- Circuit Board and Box
- 4-Lead In-line Plug (convenient to allow for disconnecting power supply from interface box)